**https://www.appliedaicourse.com/course/11/Applied-Machine-learning-course**

**import numpy np**

arr.random.rand()

arr.random.randn()

arr.random.random\_sample()

arr.random.randint()

arr.range()

arr.shape()

arr.reshape()

arr.arange()

np.zeros\_like()

np.random.choice()

[**Panasonic Carrier on DS**](https://www.careerex.in/campaign/datascience/?utm_source=SW&utm_medium=YT_CPV&utm_campaign=DS&gclid=Cj0KCQiAzZL-BRDnARIsAPCJs73G6nrvPP5M-hRuZUhYyLuBrSFNV0tMdBE_XJNoV23DgoQ2Lw7knoUaAp5OEALw_wcB)

Import sweetviz DSProj 09

Import pickle DSProj10

Import streamlit DSProj10

Import pandas\_profiling DSProj11

From pandas\_visual\_analysis import VisualAnalysis DSProj12

Import vaex DSProj13

Import dtale DSProj14

**import pandas pd**

df.head()

df.tail()

df.loc()

df.iloc[:,:]

df.iloc[:,:].values -- convert df to arrays

df.to\_csv()

df[].values\_counts()

df.isnull.sum()

df[].value\_counts()

df[].unique()

pd.read\_csv()

df.info()

df.describe()

df.corr()

pd.read\_html()

pd.read\_excel()

pd.read\_pickle()

pd.apply()

pd.get\_dummy()

pd.concat()

df.shape()

pd.pandas.set\_option()

**import matplotlib.pyplot as plt**

plt.scatter(x,y,c='g')

plt.xlable('X axis')

plt.ylable('Y axis')

plt.title('Graph in 2D')

plt.savefig()

plt.plot()

plt.subplot()

plt.bar()

plt.hist()

plt.boxplot()

plt.pie()

plt.show()

**import seaborn as sns**

sns.set\_style()

for continues data

distplot - univariate analysis

joinplot - bivariate analysis

pairplot - multivariate analysis

sns.heatmap(df.corr())

sns.jointplot()

sns.pairplot()

sns.distplot()

for categorical data

boxplot

violinplot

countplot

barplot

sns.countplot()

sns.barplot()

sns.boxplot()

sns.violinplot()

sns.facetgrid()

pip install xgboost

from wordcloud import wordCloud, STOPWORDS

from sklearn.ensemble import RandomForestRegressor, RandomForestClassifier

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import r2\_score,mean\_squared\_error,mean\_absolute\_error,accuracy\_score,classification\_report,confusion\_matrix,roc\_auc\_score,roc\_curve

from sklearn.preprocessing import StandardScaler, MinMaxScaler, OneHotEncoder,LabelEncoder

from sklean.model\_selection import RandomizedSerchCV, StratifiedKFold

from sklearn.calibration import CalibratedClassifierCV

from sklean.svm import LinearSVC

from sklean.neighbors import KNeighborsClassiifier

from sklean.naive\_bayes import GaussianNB

from skopt import BayesSearchCV

import xgboost as xgb

xgb.XGBRegressor(objective ='reg:squarederror', learning\_rate = 0.1, max\_depth = 30, n\_estimators = 100)

Least Sum of Square (LSS) = min

Mean Absolute Error (MAE) =

Mean Square Error (MSE) =

Root Mean Square Error (RMSE) = represents the standard deviation of the residual (ie., differences between the model prediction and the true values (training data))

Mean Absolute Error Percentage (MAPE) =

Mean Percentage Error (MPE) =

R Square – Coefficient of Determination =

Adjusted = 1 -

**Parameter:**

* Values that are obtained by the training process such as network weights and biases

exp: In Linear Regression y=mx +b, in this, model is trying to get the value of m, b by training

**Hyperparameter:**

* Values set prior to the training process such as number of neurons, layers, learning rate etc.

exp: In regression model setting up ‘learning rate’ to reach ‘Global Minimum’ (least error)

**Batch Size**

* Batch size indicates the number of samples that will propagate through the algorithm

Exp: Let’s assume that we have 1000 images for training. For batch size = 50, the first 50 images (from index 1 to index 50) will be propagated to the training algorithm and used for training. Then the next 50 images are propagated (index 51 to index 100). Procedure is repeated until we use all the training data.

* If the batch size is small, ML models can easily escape local minimum areas
* If the batch size is large, ML model can get stuck in a local minimum.

**Hyperparameters Optimization:**

- Grid Search

* GridSearch preforms exhaustive search over a specified list of parameters

Note that you will have the following number of combinations 3\*3\*3\*2=54

We will run each combination 5 times since we set the cross validation =5

Total number of runs = 54\*5 = 270

parameter\_grid = {‘max\_depth’:[3,6,10],

’learning\_rate’:[0.01,0.05,0.1], ‘n\_estimators’:[100,500,1000],

‘colsample\_bytree’:[0.3,0.7])

grid = GridSearchCV(estimator = model,

param\_grid = ‘neg\_mean\_squared\_error’,

cv = 5,

verbose = 5)

- Randomized Search

* Grid search works great if the number of combinations are limited
* In scenarios when the serach space is large, RandomizedSearchCV is preferred.
* The algorithm works by evaluating a select few numbers of random combinations

grid = { ‘n\_esstimaters’: [100,500,900,1100,1500]

‘max\_depth’: [2,3,5,10,15]

‘learning rate’:[0.05,0.1,0.15,0.20]

‘min\_child\_weight’:[1,2,3,4]

‘booster’:[‘gbtree’,’gblinear’])

random\_cv = RandomizedSearchCV(estimator=model,

param\_distribution = grid,

cv =5,

n\_item = 50,

scoring = ‘neg\_mean\_obsolute\_error’,

verbose = 5,

return\_train\_score = True)

- Bayesian Optimization

* Bayesian optimization overcomes the drawbacks of random search algorithms by exploring search spaces in a more efficient manner
* If aa region in the search space appears to be promising (ie., resulted ina small error), this region should be explored more which increases the chances of achieving better performance.

search\_space = (‘max\_depth’:[4,20]

‘n\_estimators’:[100,500],

‘learning\_rate’:[0.01,1.0,’log-uniform’])

xgb\_bayes\_search = BayesSearchCV(model,

search\_space,

n\_iter =50,

scoring = ‘neg\_mean\_absolute\_error’,

cv=5)

xgb\_bayes\_search.fit(X\_train,y\_train)

**EDA03:**

Linear Regression

1. Logistic Regression 2. Linear Regression

Non-Liner Regression

1. Decision Tree 2. Random Forest 3. Xg Boost 4. SVM 5. Ada Boost 6.KNN

**Stat01:** Z – Score Statistics

To convert the Gaussian Normal Distribution to Standard Normal Distribution use Z-score formula Z – Score = (xi - μ) / σ

Note: For Standard Normal Distribution Mean μ is always 0 and SD σ always variance of 1

**Stat03:** [Linear Regression Math Intuition](https://www.youtube.com/watch?v=1-OGRohmH2s&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=30)

Equation for straight line is y = mx + c m = slope/co-efficient c = intercept

Prediction =

[**Cost Function**](file:///Users/francispaulraj/Training/Krish/image/Stat03_6_53.png) =

m = Total Number of points

= Line of Prediction

= Points away from prediction line

* When Cost Function is Zero (0) then, it tells all the points falls on the Regression Line

[**Gradient Descent**](file:///Users/francispaulraj/Training/Krish/image/Stat03_15_44.png)

Gradient descent is an optimization algorithm used to find the value of parameters (coefficients) of a function (f) that minimizes a cost function. Common examples of algorithms with coefficients that can be optimized using gradient descent are Linear Regression and Logistic Regression.

**Global Minima**

In Gradient descent for optimization, to get optimal slop (coefficient) by using convergence theorem, the point it converges from -ve slop to +ve slop is called Global Minima.

Global Minima is the point where the truth is closer to the prediction

**Convergence Theorem**

L = Learning Rate

= Slope

**DSReg02:** is central metric introduced for linear regression

SSres = SSmean =

value range from zero to one (0 to 1), Higher the value Greater the model

SSmean (MSE) Mean Squared Error is to calculated the difference between each target y and the model’s predicted value (i.e., the residual)

**Stat04:** [**RIDGE and LASSO Regression:**](https://www.youtube.com/watch?v=9lRv01HDU0s&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=31)

Ridge and Lasso regression helps the models to reduce overfitting (Low Bias, High Variance) model to Low Variance, Low Bias

* In Ridge regression the slope value will tend to move towards to zero (0) value.
* In Lasso regression it helps to reduce the overfitting model as well as for multiple feature analysis, it helps for feature selection. In Lasso regression, when slope value is closer to zero (0) we can drop this feature and use rest of the feature for further analysis
* Lasso regression (L1 regression) can be useful if we have several independent variables that are useless
* Ridge regression can reduce the slope close to zero (but not exactly zero) but Laasso regression can reduce the slope to be exactly equal to zero.
* **When to choose L1:** If you believe that some features are not important and you can afford to lose them, then L1 regularization is a good choice. The output might become sparse since some features might have been removed.
* **When to choose L2:** If you believe that all features are important and you’d like to keep them but weigh them accordingly.

[Ridge Regression](file:///Users/francispaulraj/Training/Krish/image/Stat04_3_0.png) =

[Lasso Regression](file:///Users/francispaulraj/Training/Krish/image/Stat04_17_05.png) =

**DSReg03:** [**Multicollinearity**](https://www.youtube.com/watch?v=NAPhUDjgG_s&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=34)

In a regression model if features are correlated (>90) each other then it is called Multicollinearity.

In this type of model, we can take one feature among the corelated the features, an can drop the remaining.

If features are highly correlated (>90) then feature or can take for regression model.

**DSReg04:** [**Bias & Variance Overfitting Underfitting**](https://www.youtube.com/watch?v=BqzgUnrNhFM&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=35)

*Underfitting:* In Training data set, the model gives more error, then the model is called Underfitting.

In Underfitting, the accuracy will be low for both Training and Test data set, and because of more error It has High Bias for Training data set, High variance for Test data set.

High Bias, High Variance

*Overfitting:* In Training data set, all the data are covered by the model, then the model is called Overfitting. This kind of model will give more error (variance) for Test data set. In Overfitting, accuracy will be high for Training data but accuracy will be low for Test data. In Overfitting, because of error is low for Training data set, it has Low Bias for Training data set, and High Variance for Test data set.

Low Bias, High Variance

Note: Bias meaning error of Training dataset, Variance meaning error of Test dataset

Binary Tree Classifier is example for Over Fitting, Random Forest Classifier is example for Low Bias, Low Variance.

* Best model will have Low Bias and Low Variance
* When our error is biased, it means the model’s prediction is consistently far away from the actual value
* This could be a sign of poor sampling and poor data
* One objective of a biased model is to trade bias error for generalized error. We prefer the error to be more evenly distributed across the model.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Underfitting** | High Bias – High Error – Training Data | **Overfitting** | Low Bias – Low Error. – Training Data | **Bestfitting** | Low Bias |
| High Variance – High Error. – Test Data | High Variance – High Error. – Test Data | Low Variance |

**DSReg05**: [**R Square and Adjusted R square**](https://www.youtube.com/watch?v=WuuyD3Yr-js&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=36)

gives goodness of best fit in regression model, with range from 0 to 1, and closer to 1 is best fit.

,

= sample R square, p = Number of predictors (independent features), N = Total sample size

Why is used, when we use multiple features ( ) the co-efficient of each feature will tend to increase value, including the feature which does not corelate with output, which will lead into wrong prediction. To avoid such errors, we use .

When number of features increases and corelated with output then the value will increase little than ,but when features are not corelated with output then the will degrees than

Difference of and

* Every time you add an independent variable to a model, then increases, even if the independent variable is insignificant, it never declines. Whereas increases only when independent variable is significant and affects dependent variable.
* value always be less than or equal to value.

**Sagemaker:**

* One limitation of is that it increases by adding independent variables to the model which is misleading since some added variables might be useless with minimal significance.
* Adjusted overcomes this issue by adding penalty if we make an attempt to add independent variable that does not improve the model
* Adjusted is a modified version of the and takes into account the number of predictors in the model
* If useless predictors are added to the model, Adjusted will decrease
* If useful predictors are added to the model, Adjusted will increase
* K is the number of independent variables and n is the number of samples = 1 -

**Stat06: T test, CHI square test, ANOVA test when to use what?**

* In a given data set of having multiple categorical column, for ONE categorical feature use ONE proposal test.
* For TWO categorical feature use CHI square test.
* For ONE continuous (numerical) feature use T test
* For TWO continuous (numerical) feature use correlation
* For continuous (numerical) feature and ONE categorical feature or TWO categorical feature with more than one category use ANOVA test.

**Stat07: P-Value, T test, Correlation**

A t-test is a type of inferential statistic which is used to determine if there is significant difference between the means of two groups which may related in certain feature. It has two types

1. one sampled t-test 2. Two sampled t-test

One sample t-test: The test will tell us whether means of the sample and the population are different

where

Where

μ = proposed constant for the population mean

= sample means

n = sample size (number of observations)

s = sample standard deviation

= estimated standard error of the mean

Two sample t-test: The independent samples t test or two sample t test compares the means of two independent groups In order to determine whether there is statistical evidence that the associated population means are significantly different. The independent samples t test is a parametric test. This test is also known as independent t test.

**Stat08:** **Chi Square test**

The test is applied when you have two categorical variables from a single population. It is used to determine whether there is a significant association between the two variables.

**Stat09:** [**Performance Metrics for Classification Problem in ML**](https://www.youtube.com/watch?v=aWAnNHXIKww&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=41)

In classification problem especially for Binary classification if the data is unbalanced (ie., 80% - 20% or 90% - 10%) the ML model for Binary classification will be biased on the higher percentage. To come out of this problem we use Confusion Matrix

|  |  |  |  |
| --- | --- | --- | --- |
| Predicted | Actual Value | | |
|  | 1 | 0 |
| 1 | TP | FP |
| 0 | FN | TN |

FP = False Positive (FPR, Type 1 Error, Specificity)

FPR Ratio =

FN = False Negative (FNR, Type 2 Error)

Accuracy =

Misclassification rate (Error Rate) =

Recall (TPR, Sensitivity) – Out of total actual +ve value, how many +ve value predicted correctly (When the class was actually TRUE, how often did the classifier get it right). Whenever FN is much important then use Recall.

Precision (+ve Predicted Value) – Out of the total Predicted +ve results, how many results are actual +ve (When the model predicted TRUE class, how often was it right) . Whenever FP is much important then use Precision

F βeta Score:

We have to choose score when FP and FN are more important, in this we have to choose the value of β. The value of β may change depends on FP and FN.

Exp:

In confusion matrix if both FP and FN are equally important then select β =1.

when β =1 then Which gives Harmonic Mean = .

Exp:

In confusion matrix if FP (Type one Error) is more important than FN (Type two error) then select β = 0.5

when β =0.5 then

Exp:

In confusion matrix if FN (Type two error) is more important than FN (Type one error) then increase the β value ie., β=2

when β =2 then

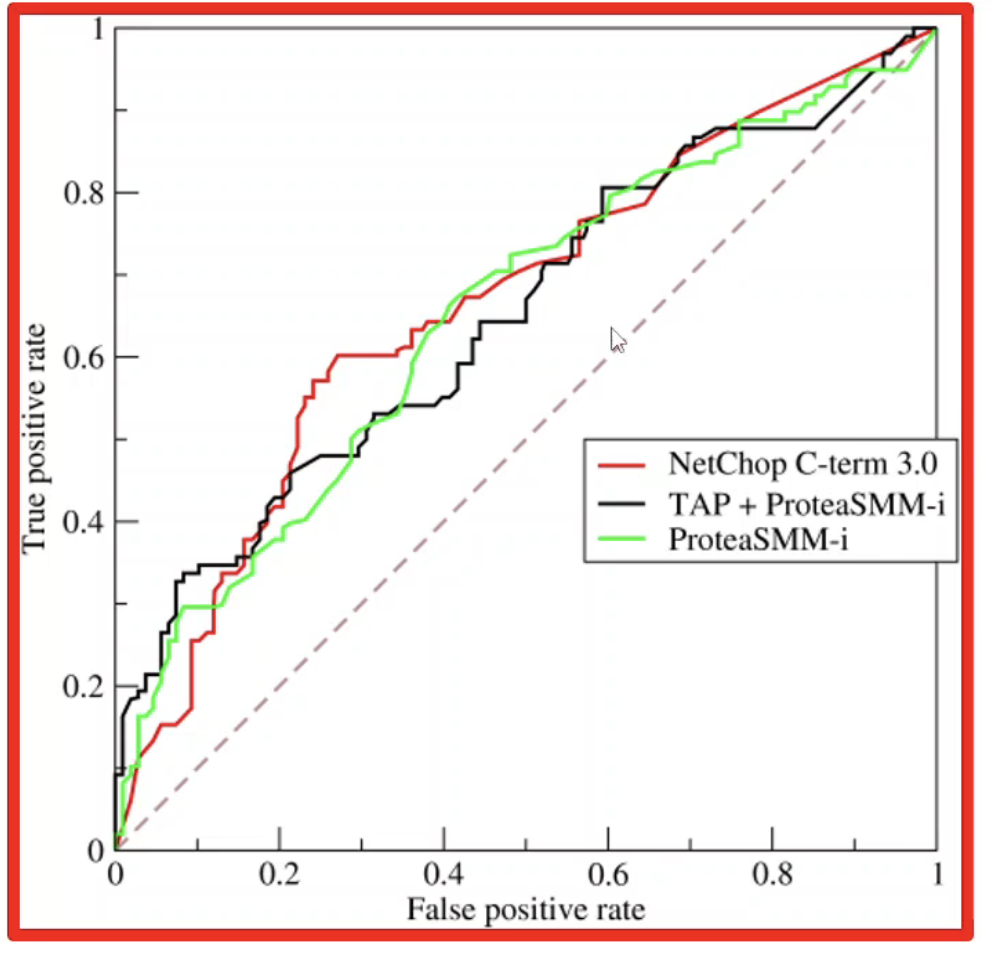
Score Summary:

β =1 for FP (Type one error, Precision), FN (Type two error, Recall) has higher impact.

β =0.5 for FP (Type one error, Precision) has higher impact than FN (Type two error, Recall)

β =2 for FN (Type two error, Recall) has higher impact than FP (Type one error, Precision)

When FP & FN have greater impact then select β =1

When FP has more impact then reduceβ =0.5

When FN has more important than increase β > 2

ROC (Receiver Operation Characteristic) – ROC curve is created by plotting TP against FP at various models’ threshold

The true-positive rate is also known as sensitivity, recall or probability of detection in machine learning.

The false-positive rate is also known as the probability of false alarm and can be calculated as (1-specificity)

ROC Helps to determine best threshold value

AUC (Accuracy Under the Curve) – Summarizes the impact of TPR and FPR is a single value

AUC Helps to determine which classification model is best

**DSReg06: Logistic Regression**

Reasons not to use Linear Regression for Binary Classification

1. Whenever we have lot of outliers the best fit line completely deviated
2. In Binary classification, most of the output will be >1 or < 0

**DSReg06:** [**Logistic Regression**](https://www.youtube.com/watch?v=uFfsSgQgerw&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=43)

* Usually Logistic Regression is mostly used for Binary classification, with some Hyper tuning it can be used for Multiclass classification.
* In Binary classification, the Logistic Regression is used with the intuition that it can linearly separate this two classification groups.
* In Logistic regression, Cost function should be always maximum ie., where is co-efficient of feature . which will give the distance of from the plane or linear straight line.
* In logistics regression for binary classification, when there is an outlier it tent to increase the value (ie., ), which is wrong, to avoid this we use sigmoid function where z is The sigmoid function range from 0 to 1.
* Regression – output contains continuous numeric values
* Binary classification – output label must be either 0 or 1
* Multiclass classification – output label must be from 0 to num\_class -1

**SageMaker Linear Learner:**

**Preprocessing:**

* Normalization or feature scaling is offered by Linear Learner
* Feature scaling is a critical preprocessing step to ensure that the model does not become dominated by the weight of a single feature

**Training:**

* Linear Learner uses stochastic gradient descent to perform the training
* Select an appropriate optimization algorithm such as Adam, AdaGrad
* Hyperparameters can be selected and turned (exp: learning rate)
* Overcome model overfitting using L1, L2 regularization

**Validation:**

Trained models are evaluated against a validation dataset and best model selected baased on the following metrics:

* For regression: mean square error, root mean square error, absolute error
* For classification: F1 score, precision, recall, or accuracy

**DSReg07:** [**Logistic Regression Multiclass classification**](https://www.youtube.com/watch?v=V8fS0T_ktn4&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=44) **(One Vs Rest)**

We can use Logistic regression for Multiclass classification by making one feature with +ve label and rest of the features as -ve label.

**DSDTree01: Decision Tree**

In the Binary classification (Decision Tree), it is important to identify the feature to begin splitting of nodes in decision tree, Entropy helps to measure the purity of the splits to reach the leaf node quickly from root node.

where are Probability with % of +ve and % -ve class

Exp: node has then split with the Entropy

* Entropy value ranges between 0 to 1, lesser the value better the Entropy

**DSDTree02:** [**Information Gain**](https://www.youtube.com/watch?v=FuTRucXB9rA&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=46)

Case 1

Case 2

)

F1 Root Node -> = 9Y,5N

F2  Leaf Node (sub set) -> ) = 6Y,2N (Pure split)

F3  Leaf Node (sub set) -> ) = 3Y,3N (Impure split)

Total sample

Sample after the split

We get the following values after computing these values in

for F1 = 0.94

for F2 = 0.81

for F3 = 1

Information Gain (s, F1) = ) -

Gain (s, F1) = 0.94 - 0.049

For case1 we got information gain of 0.049, The information gain logic will calculate for case2 and so on until it gets the highest value. The root feature in split structure which gives the highest information gain value will consider for start node (Root Node) in binary classification.

**DSDTree03:** [**Gini Impurity**](https://www.youtube.com/watch?v=5aIFgrrTqOw&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=47)

Gini Impurity also helps to find purity of the split in binary classification like Entropy, but most of time Gini impurity is better than Entropy because of computational performance (No computational time for log)

**DSDTree04:** [**Decision Tree Split for Numerical Feature**](https://www.youtube.com/watch?v=5O8HvA9pMew&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=48)

Consider the following when use Decision Tree split for Numerical Feature

1. Sorting of Numerical Feature
2. Create Threshold value for Numerical Feature

Disadvantage of using Decision Tree split for Numerical Feature is, If the number of Numerical Feature is large, it takes longer time for computing Gini or Entropy impurity and Information Gain for each and every threshold value.

**DSProj01: Advance House Price Prediction**

Life cycle of Data Science project

1. Data Analysis
2. Feature Engineering
3. Feature Selection
4. Model Building
5. Model Deployment

Data Analysis

1. Missing Values
2. All the numerical variables
3. Distribution of numerical variables
4. Categorical variables
5. Cardinality of categorical variable
6. Outlier
7. Relationship between independent and dependent feature (sales Vs Price)

**DSProj03: Advance House Price Prediction – Feature Engineering**

Steps used in Feature Engineering for this model

1. Missing values
2. Temporal variable
3. Categorical variable: remove rare labels
4. Standardize the values of the variables to the same range

**DSDTree05:** [**Performance Metrics ROC, AUC for Binary**](https://www.youtube.com/watch?v=A_ZKMsZ3f3o&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=54) **Classification**

ROC, AUC curve mostly used for Binary classification problems, which helps the SME’s and Data Scientist to decide about the performance of the model. To create ROC, AUC curve it used TPR (True Positive, Recall, Sensitivity) TPR = and FPR (False Positive, Type 1 Error) .

TPR – Y axis, FPR – X- axis, More the area under the cure is better the Model

**DSDTree06:** [**KNN Classification**](https://www.youtube.com/watch?v=otolSnbanQk&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=56)

Points to follow when we introduce new points to classify in KNN classification

1. Select K value, which mean how many points to select as a neighbor of the newly introduced point.
2. Calculate the distance between the newly introduced point to the K neighbor points
3. Calculate how may points (K neighbor points) closed to newly introduced point
4. The newly introduced point will be assigned to the category, in which sum of number of closed points is high to the newly introduced point.

How to find the distance between points

1. Euclidian Distance
2. Manhattan Distance

* KNN will be impacted when the data set is biased (100yes, 10no)
* KNN will be impacted when data set has outliers.
* KNN with regression the newly introduced point will be calculated by the mean of neighbor points

**DSDTree08: Ensemble, Bagging, Bootstrap Aggregation**

Ensemble means combining more than one models or technique. In classification problem Ensemble have two methods 1. Bagging 2. Boosting

|  |  |
| --- | --- |
| **Bagging (Bootstrap Agg)** | **Boosting** |
| Random Forest | ADA Boost |
|  | Gradient Boosting |
|  | XgBoost |

In the bellow figure sample data from data set will be distributed to different models with ‘Row Sampling with Replacement’ technique. Row sampling with Replacement means, It get sample data from data set and send (Boost) to model 1, when it send sample data to model 2, it sends send’s some of the data from the previous data (ie., to model1). It’s kind of overlapping at the edge of sampled data. In aggregation, It takes the highest value out of classification model, but in the case of regression (continues value) it takes the mean of the model’s output.

**Data**

**Mo4el 4**

**Model 3**

**Model 2**

**Model 1**

Boost

Aggregation for highest values from models

**DSDTree10: Handling Imbalanced Dataset – Under Sampling**

In a data set for classification model, if the depended feature is in the ratio of 900YES, 100 No this model will give biased result. In order to overcome this, while sampling take 100 No and 100YES randomly is called Under Sampling. We may not get good result if Data Set size if small, it’s good to perform Under sampling technique when the Data set is big.

**DSDTree11: Handling Imbalanced Dataset – Over Sampling**

In a data set for classification model, if the depended feature is in the ratio of 900YES, 100 No this model will give biased result. In order to overcome this, while sampling take 100 No and rise this value ~ 900No, so that ratio would looks like 1:1 , this technique is called Over Sampling. In real time Over Sampling is the best solution for handling Imbalance Dataset, because in this technique we are not losing any data.

**DSDTree11: Hyperparameter Optimization for Xgboost**

Hyperparameter Optimization is very important task for any ML techniques, reason is it helps the model to use correct parameter.

How Xgboost Hyperparameters are selected / optimized by using Hyperparameter Optimizer like Radom Search, Grid Search, Bayesian Optimizer

**DSDTree13:** [**AdaBoost**](https://www.youtube.com/watch?v=NLRO1-jp5F8&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=63)

AdaBoost follows the steps

1. Create Sample weight where is total rows in the dataset.
2. Create base learns to find total error, In AdaBoost all base leaners are decision tree. Create decision tree with one depth (one parent node with two leaf nodes (stumps)) and get the entropy or Gini coefficient value for purity of splits.
3. Performance of Stump
4. Update Weight for incorrect points, ie., increase the weight of incorrect points
5. Update Weight for correct points, ie., decrease the weight of correct points

**DSUnspr01:** [**K Means Clustering**](https://www.youtube.com/watch?v=AWKCCK5YHsE&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=66)

Points observed in K Means Clustering

1. Select K value (centroid value)
2. Initialize the centroid randomly
3. Select the group and find the mean of the group
4. The centroid will move into the group
5. Do the process from 2 to 4 until there is no movement between the groups
6. The distance between the centroid value is calculated by Euclidean distance method

* Elbow method is used to select K value. When compute the K value (1 to 20) on , for K=1, value will be high. When compute further (K=2, K=3, …) the value will be decreasing on certain computation for K value, will not change, that point of K value is optimized for to use in the model (K Means Clustering). The graph will look like Elbow.

Within cluster sum of square

**DSUnspr02:** [**Hierarchical Clustering**](https://www.youtube.com/watch?v=0jPGHniVVNc&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=67)

* Hierarchical Clustering is one of the Clustering methods by grouping closed points within a large cluster and plot on Dendogram.
* Dendogram is graphical representation of smallest, nearest grouping within big cluster.
* Dendogram helps to select to find number of clusters to be used in the model.

**DSUnspr02:** [**Density Based Spatial Clustering of Applications with Noise (DBSCAN)**](https://www.youtube.com/watch?v=C3r7tGRe2eI&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=68)

Points to take in DBSCAN

1. Epsilon, radius of a referenced point
2. Minimal Points, set how many points should be present in the circular area of the referenced point (say point A), If Minimal point is 4, then 4 neighbor points should present in the circular area of the referenced point, say point A.
3. Core Points, If the circular area of the referenced point (A point) contains >= Minimal Points, then the referenced point is called Core Point
4. Border Points, If referenced point say point C is not contains the number of points (< ) set from Minimal Points, but it contains at least one CORE Point (say A) within its circular area, then this point (C point) is called Border Point.
5. Noise Point, if referenced point say point D is not satisfy Minimal Points ( < Minimal Point) and doesn’t contain any Core Point within the circular area, then this point is called Noise Point. Noise Point is nothing but outlier.

Advantages of DBSCAN

* Is great at separating clusters of high density versus clusters of low density within a given datset
* Is great with handling outliers within the dataset

Disadvantages of DBSCAN

* Does not work well when dealing with clusters of varying densities, While DBSCAN is great at separating high density clusters from low density clusters, DBSCAN struggles with clusters of similar density.
* Struggles with high dimensionality data. I know, this entire article I have stated how DBSCAN is great at contorting the data into different dimensions and shapes. However, DBSCAN can only go so far, if given data with too many dimensions, DBSCAN suffers.

**Stat11: Curse of Dimensionality**

When training a model (Regression or Classification), increase in dimension (feature) will give better accuracy, but certain threshold point increase in dimension will start decrees in accuracy, this point of decrees in accuracy is called curse of dimensionality.

**DSUnspr04: Dimensional Reduction Principal Component Analysis**

Principal Component Analysis (PCA) is a unsupervised ML algorithm, which helps to reduce the number of feature in the dataset.

* To reduce the features from data set, we have to process the data through Standard Normal Distribution , were σ =1, μ=0 (use sklearn scalar sd)
* Rescale all the numerical value, so that all the value will be nearer to the PCA line. (use sklearn.preprocessing import StandardScalar)
* Apply PCA algorithm
* Apply ML algorithm

**Stat12:** [**What is Cross Validation and its types?**](https://www.youtube.com/watch?v=7062skdX05Y&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=73)

After train a model with Training dataset, we perform accuracy of the model by using Test dataset. The accuracy may change due to Train-Test dataset split ratio (70:30) and while changing number of random sample test.

Cross Validation allows us to compare different machine learning models and get a sense of how well they will work in practice OR is to test the model’s ability to predict new data

Types of Cross Validation

1. Leave One Out CV (LOCV) –
   * Say out of 1000 records, it takes ONE dataset as Test and Remaining as Train, this task will happen for 1000 records
   * Generate several models on different cross sections of the data
   * Measure the performance of each
   * Take the mean performance
   * Disadvantage – Needs high processing power due to Train-Test split for all records, leads into low bias
2. K Fold CV
   * Split the data into K group
   * Train the model on all segments except one
   * Test model performance on the remaining set
   * If K=5, split the data into five segments and generate five models
   * Disadvantage – unbalanced dataset gives more biased output
3. Stratified CV
   * Overcome the problem from K – Fold CV
   * It makes sure the number of Yes and No are equal ration in both Test-Train
4. Time Series CV

**Stat13:** [**Bayes Theorem – Conditional Probability**](https://www.youtube.com/watch?v=71oNiqPoKD8&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=75)

Conditional Probability , Probability of P(B) given that P(A) is already occurred.

**Stat14: Naïve Bayes Classifier**

From Bayes Theorem , Say we have a dataset of with output value , were namely features like . When fit these values in Bayes Theorem we get

=

,

were

y = argmax

**Stat15:** [**How to apply Naïve Bayes Classifier on Text Data (NLP)**](https://www.youtube.com/watch?v=temQ8mHpe3k&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=77)

It’s always good to use Naïve Bayes classifier for NLP

**Stat16:** [**Support Vector Machine (SVM)**](https://www.youtube.com/watch?v=H9yACitf-KM&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=78)

SVM helps to solve both Regression and Classification problems

* ***Support Vectors***– +ve or -ve points that helps to create marginal planes parallel to Hyperplanes are called Support Vectors
* ***Hyperplanes*** – Hyperplane is a regression line/plane that separates the groups (+ve, -ve) of datasets in classification problem
* ***Marginal Distance*** – SVM makes sure to create two more hyperplanes that is parallel to Hyperplane, and also it passes through the nearest points of +ve, and -ve from Hyperplane. The distance between Hyperplane to +ve, -ve Hyperplane is called Marginal Distance.
* Idea of creating Hyperplane is to find plane that separates the +ve/-ve point with maximum distance. We can create more than one Hyperplane for a single dataset, but the best Hyperplane will have maximum Marginal Distance.
* ***Linear Separable*** – dataset in classification problem, +ve or -ve groups are separated linearly with Hyperplane (simple straight line)
* ***Non-Linear Separable*** - dataset in classification problem, +ve or -ve groups are not separated linearly is called Non-Linear Separable, but this kind of problem can be solved by SVM Kernel.
* ***SVM Kernel –*** SVM Kernel is used to separate the +ve and -ve groups that are non-linear, by converting (or creating another dimension) 2D to 3D space, which helps to create Hyperplane between +ve, -ve groups.

**DSDTree14:** [**Gradient Boosting**](https://www.youtube.com/watch?v=Nol1hVtLOSg&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=80)

1. Create Base Model

2. Find residual value

3. Construct Decision Tree

Gradient Boosting => Base Model -> Decision Tree 01 -> Decision Tree 02 .. -> Decision Tree n

**DSDTree15:** [**Gradient Boosting**](https://www.youtube.com/watch?v=Oo9q6YtGzvc&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=81) **-Part2**

|  |  |  |
| --- | --- | --- |
| Exp | Degree | Salary |
| 2 | BE | 50K |
| 3 | Phd | 70K |
| 4 | Master | 60K |

Required Task for Gradient Boosting Algorithm

1. Provide Input and Output value (Input values – Exp, Degree, Output values – Salary)
2. Provide loos functions (for Regression – RME, MSE, for Classification – Hinge)
3. Provide how many Binary Tree needed
4. Sudo Algorithm
   * Initialize Model with constant value, for the base model.
   * First order derivative for , equation of lose function
   * Iterate M =1 to M
   * Compute Pseudo residuals
   * Fit a base learner where input values are were is difference between (salary) output value and constant value.

**DSDTree16:** [**XgBoost (Extreme Boost) Classification**](https://www.youtube.com/watch?v=gPciUPwWJQQ&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=82)

**DSDTree17:** [XgBoost Regression](https://www.youtube.com/watch?v=w-_vmVfpssg&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=83)

**What is Boosting:**

* Boosting works by learning from previous mistakes (errors in model predictions) to come up with better future predictions
* Boosting is an ensemble machine learning technique that works by training weak models in a sequential fashion
* Boosting algorithms work by building a model from the training data, then the second model is built based on the mistakes (residuals) of the first model. The algorithm repeats until the maximum number of models have been created or until the model provides good predictions.

**What is Ensemble Learning:**

* XGBoost is an example of ensemble learning
* Ensemble techniques such as bagging and boosting can offer an extremely powerful algorithm by combining a group of relatively weak/average ones
* For example, you can combine several decision trees to create a powerful random forest algorithm
* Boosting can reduce variance and overfitting and increase the model robustness

**Advantages:**

* No need to perform any feature scaling
* Can work well with missing data
* Robust to outliers in the data
* Can work well for both regression and classification
* Computationally efficient and produce fast predictions

**Disadvantages:**

* Poor extrapolation characteristics
* Need extensive tuning
* Slow training

**Gen04: Step by Step to lean ML**

1. Understand the math behind algorithms

2. How these algorithms behave w.r.t. numerical and categorical variables?

a. Decision Tree uses different ways to split numerical and categorical predictors/variables.

3. How these algorithms work with Text Data?

a. Stemming and Lemmatization

b. Bag of words c. TF-DIF d. Word2Vec

4. For which scenario these algorithms are used?

a. Regression - Linear regression

b. Classification - Logistic regression, Naive Bayes Classifier

c. Both - Decision Tree

5. Over-fitting and Under-fitting Conditions: -

a. Hyper parameter tuning

b. Decision Tree Pruning

6. Impact of Algorithm w.r.t Imbalanced Datasets and how do you fix that?

a. Binary Classification Problem

b. Feature Scaling # Up-sampling # Down-sampling

7. Impact of Outliers, how to treat them?

8. For which Algorithm, feature Scaling/Normalization is required w.r.t Datasets: -

a. DT, Random Forest, XGBoost, Gradient Boosting, ADABoost - Not required

b. Linear Regression, Logistic Regression - Required.

**StatML01: How to learn Statistics for Data Science**

Basic Stats

1. Introduction to Basic Term
2. Variables
3. Random Variables
4. Population, Sample, Population Mean, Sample Mean
5. Population Distribution, Sample Distribution, and Sampling Distribution
6. Mean, Median, Mode
7. Range
8. Measure of Dispersion
9. Variance
10. Standard Deviation
11. Gaussian / Normal Distribution

Intermediate Stats

1. Standard Normal Distribution
2. Z score
3. Probability Density Function
4. Cumulative Distribution Functions
5. Hypothesis Testing
6. Many Different Plotting graphs
7. Kernel Density Estimations
8. Central Limit Theorem
9. Skewness of Data
10. Covariance
11. Pearson Correlation Coefficient
12. Spearman Rank Correlation

Advanced Stats

1. Q – Q Plot
2. Chebyshev’s Inequality
3. Discrete and Continuous Distribution
4. Bernoulli and Binomial Distribution
5. Log Normal Distribution
6. Power Law Distribution
7. Box Cox Transform
8. Poisson Distribution
9. Application of Non-Gaussian Distribution

**StatML05: Covariance**

Covariance

Variance

**StatML06: Mean, Median, Mode**

Mean μ =

**StatML07: Population Vs Sample Mean**

Population Mean μ =

Sample Mean

**StatML08: Random Variables and tis Types**

Random Variable

* Normal Random Variable
  + Discrete Random Variable
    - Exp: Bank Account Number, Number of Student in Class
  + Continuous Random Variable
    - Exp: Salary, Height
* Categorical Random Variable

**StatML09: Gaussian, Normal Distribution**

Empirical Formula

**StatML11: Chebyshev’s InEquality**

If a random variable are falls in Gaussian Distribution ) then you get

If a random variable are not falls in Gaussian Distribution ) , then you get

**StatML12: Pearson Correlation Coefficient, Correlation and Covariance**

Covariance is , Helps to find the direction of the relationship.

Pearson Correlation Coefficients , Helps to find the Strength and the Direction of the relationship, and the range between -1 to +1.

**StatML14: Outliers:**

* What are the criteria to identify an outlier?
* Data point that falls outside of 1.5 times of an interquartile range above the 3rd quartile and below the 1st quartile
* Data point that falls outside of 3 standard deviations, we can use a z score and if the z score falls outside of 2 standard deviation
* What is the reason for an outlier to exists in dataset?
* Variability in the data
* An experimental measurement error
* What are the impacts of having outliers in a dataset?
* It causes various problems during our statistical analysis
* It may cause a significant impact on the mean and the standard deviation
* Various ways of finding the outlier
* Using scatter plots
* Box plot
* Using Z score
* Using IQR

**StatML15: Standardization Vs Normalization:**

* What is magnitude and Unit: ( exp: age = 25 years, here 25 is magnitude and years is unit)
* Normalization (min -max Normalization) helps to scale down the feature in the dataset

between 0 to 1

* Standardization (Z – Score Normalization), Here all the features will be transformed in such a way that It will have the properties of a standard normal distribution with mean (μ) =0. And standard deviation (σ)=1.
* Standardization scaling mostly used technique for scaling features (Exp: Linear Regression, KNN, Clustering)
* Normalization scaling mostly used in CNN, ANN where the datasets are image and video.
* No need of scaling for AdaBoost, Random Forrest, Decision Tree, because models already splits by values.
* Scaling (Standardization or Normalization) is required when we use any ML algorithm that require gradient calculation.
* In case of neural networks, normalization is preferred since we don’t assume any data distribution.
* Standardization is preferred when data follows gaussian distribution
* Standardization is preferred over normalization when there are a lot of outliers.

**When should I Perform Standardization Vs Normalization**

* Scaling (standardization or Normalization) is required when we use any ML algorithem that require gradient calculation (gradient boosting)
* Example of ML algorithms that require gradient calculations are linear/logistic regression and artificial neural network
* Having different scales for each feature will result in a different step size which in turn jeopardizes the process of reaching minimum point
* Scaling is not required for distance-based and tree-based algorithms such as K-Means Clustering, Support Vector Machines and K Nearest Neighbors, Decision Trees, Random Forest, and XG-Boost
* Standardization is preferred when data follows gaussian distribution
* Standardization is preferred over normalization when there are lot of outliers
* In case of neural networks, normalization is preferred since we don’t assume any data distribution

**FeuEng01:** [**One Hot Encoding for Multi Categorical Variables**](https://www.youtube.com/watch?v=6WDFfaYtN6s&list=PLZoTAELRMXVPwYGE2PXD3x0bfKnR0cJjN)

One Hot Encoding is used to help to convert features into dummy variables, were the feature has large number of Multi – Categorical variable

One Hot encoding of top variables

Advantages:

* Straightforward to implement
* Does not require hrs of variables exploration
* Does not expand massively the feature space (number of columns in the dataset)

Dis-Advantages:

* Does not add any information that may make the variable more predictive
* Does not keep the information of the ignored labels

Because it is not unusual that categorial variables have a few dominating categories and the remaining labels add mostly noise, this is a quite simple and straightforward approach that may be useful on many occasions.

It is worth nothing that the top 10 variables is a totally arbitrary number. You could also choose the top 5 or top 20.

**FeuEng02:** [**Different Types of Feature Engineering Encoding Techniques**](https://www.youtube.com/watch?v=OTPz5plKb40&list=PLZoTAELRMXVPwYGE2PXD3x0bfKnR0cJjN&index=2)

Types of Encoding

* Nominal Encoding

Categorical features don’t need to re-arrange the categories (ex;NJ,CA,WA)

* One Hot Encoding

Covert the categories into Dummy variables by the length of total number of unique categories minus one (N unique categories -1)

* One Hot Encoding with many Categories

Count the number of repeated categories in a feature, sort by descending order to get TOP 10 categories which are repeated the most.

* Mean Encoding
* It takes categories in a feature and its output value (A 1,B 2,A 0,D 5, D 2) and takes mean of output value for each categories, and uses its mean value for corresponding categories. (A ->0.5, B ->2,D ->3.5)
* Ordinal Encoding

Categorical feature, were we re-arrange the categories based on RANK (ex: Phd, Msc, Bsc)

* Label Encoding

Give labels in terms of Rank (ex: Phd -1 , Msc -2, Bsc -3)

* Target Guided Encoding

It takes categories in a feature and its output value (A 1, B 2, A 0,D 5, D 2)) and takes mean of output value for each categories and because its Ordinal Encoding at the end It Ranks the Mean (D 3.5 -> R3, B 2 ->R2, A .5 ->R2)

**FeuEng03: Why Do we need to perform Feature Scaling?**

Features in a dataset may have different Magnitude and Units of Discreet and Continuous value (exp: 10 years, 10 is magnitude, years unit), due to this we have to convert into Feature Scaling before applying any model. We have ‘minmax scaling (range from -1 to 1)’, ’standard scaling’ techniques. We don’t need Feature Scaling for RF, BT, AdaBoost

**FeuEng04: How to Handle Missing Values in Categorical Features?**

* Remove the Rows
* Replace the Most Frequent Values (Mode)
* Apply classifier algorithm to predict
* Apply unsupervised techniques

**FeuEng05: Feature Engineering – Handle Categorical Features Many Categories (Count/Frequency Encoding)**

**FeuEng06: How to Handle Ordinal Categories (Ordinal Encoding)**

Ordinal data is a Categorical, Statistical data type where the variables have natural ordered categories and the distances between the categories is not known.

**FeuEng07: Feature Engineering All techniques to Handle Missing Values**

What are the types of missing value?

1. *Missing Completely at Random MCAR*. A variable is missing completely at random (MCAR) if the probability of being missing is the same for all the observations. When data is MCAR, there is absolutely no relationship between the data missing and any other values, observed or missing, within the dataset. In other words, those missing data points are a random subset of the data. There is nothing systematic going on that makes some data more likely to be missing than other.
2. *Missing Data Not at random (MNAR)*. Systematic missing value. There is absolutely relationship between the data missing and any other values, observed or missing, within the dataset.
3. *Missing at random (MAR)*

All the techniques of handling missing value

1. Mean/ Median /Mode replacement

Advantages:

Easy to implement (Robust to outliers)

Faster way to obtain the complete dataset

Dis-Advantages:

Change or Distortion in the original variance

Impacts Correlation

1. Random Sample Imputation

Random sample imputation consists of taking random observation from the dataset and we use this observation to replace the nan values. It assumes that the data are missing completely at random MCAR

Advantages:

Easy to implement

There is less distortion in variance

Dis-Advantages:

Every situation randomness won’t work

1. Capturing NAN values with a new feature

Advantages:

* Easy to implement
* Captures the importance of missing values

Dis-Advantages:

* Creating additional Features, it may lead into curse of dimensionality

1. End of Distribution Imputation

Advantages:

* Easy to implement
* Captures the importance of missingness if there is one

Dis-Advantages:

* Distorts the original distribution of the variables
* If missingness is not important, it may mask the predictive power of the original variable by distorting its distribution
* If the number of NA is big, it will mask true outliers in the distribution
* If the number of NA is small, the replaced NA may be considered an outlier and pre-processed in a subsequent feature

1. Arbitrary Imputation **(FeuEng09)**

Advantages:

* Easy to implement
* Captures the importance of missingness if there is one

Dis-Advantages:

* Distorts the original distribution of the variables
* If missingness is not important, it may mask the predictive power of the original variable by distorting its distribution
* Hard to decide which value to use

1. Frequent Categories Imputation **(FeuEng09)**

* Add the most frequent category to the specific feature, do it for all the feature in a dataset
* Create new category for missing values (nan) and add to features in a dataset.
* Use random sample to fix nan value in a dataset

Advantages:

* Easy to implement
* Captures the importance of missingness if there is one

Dis-Advantages:

* Distorts the original distribution of the variables
* Since we are using more frequent labels, it may use them in an over represented way, if there are more nans.

How to Handle Categorical Features:

* One Hot Encoding **(FeuEng10)**
* Count or Frequency Encoding **(FeuEng10)**

Advantages:

* Easy to implement
* Captures the importance of missingness if there is one

Dis-Advantages:

* It will provide same weight if the frequencies are same
* Target Guided ordinal Encoding **(FeuEng10)**

Ordering the labels according to the target

Replace the labels by the joint probability of being 1 to 0

* Mean Encoding **(FeuEng10)**
* Probability Ratio Encoding **(FeuEng11)**

**FeuEng07: All Standardization and Transformation Techniques**

Types of Transformations:

* Normalization and Standardization
* Scaling to Minimum and Maximum values
* Scaling to Median and Quantiles
* Gaussian Transformation
* Logarithmic Transformation
* Reciprocal Transformation
* Square Root Transformation
* Exponential Transformation
* Box Cox Transformation

Standardization:

We try to bring all the variables or features to a similar scale. Standardization means centering the variable at zero

Use Standardscalar from sklearn library

Min Max Scaling:

Min Max Scaling scales the values between 0 to 1. X scaled =

from sklearn.preprocessing import MinMaxScalar

Robust Scalar:

It is used to scale the feature to median and quantiles. Scaling using median and quantiles consists of subtracting the median to all the observations, and then dividing by the interquartile difference. The interquartile difference is the difference between the 75th and 25th quantile.

IQR = 75th quantile – 25th quantile X scaled

from sklearn.preporcessing import RobustScalar

Gaussian Transformation:

If the dataset is not normally distribution then use Gaussian Transformation. To check whether feature is Gaussian or Normal distribution use Q-Q-Plot

* Logarithmic Transformation
* Reciprocal Transformation
* Square Root Transformation
* Exponential Transformation
* Box Cox Transformation

**FeuEng13: Handling Imbalanced Dataset – Machine Learning**

Install imbalanced library – pip install imbalanced-learn

from sklearn.ensemble import RandomForestClassifier

from sklearn.liner\_model import LogisticRegression

from sklearn.metrics import accuracy\_score,confusion\_matrix,classification\_report

from sklearn.model\_selection import KFold,GridSearchCV,train\_test\_split

Undersampling:

from imblearn.under\_samping import NearMiss

from collections import Counter

Oversampling:

from imblearn.over\_sampling import randomOverSampler

SMOTETomek:

from imblearn.combine import SMOTETomek

Ensemble Techniques:

from imblearn.ensemble import EasyEnsembleClassifier

**FeuEng14: Outlier, Skewed and Impacts on Machine Learning Usecases**

Which Machine Learning Models are Sensitive to Outliers

* Naivye Bayes Classifier – Not Sensitive to Outliers
* SVM – Not Sensitive to Outliers
* Linear Regression – Sensitive to Outliers
* Logistice Regression – Sensitive to Outliers
* Decision Tree Regressor or Classifier – Not Sensitive to Outliers
* Ensemble (RF,XGboost, GB) – Not Sensitive to Outliers
* KNN – Not Sensitive to Outliers
* Kmeans – Sensitive to Outliers
* Hierarchal – Sensitive to Outliers
* PCA – Sensitive to Outliers
* Neural Networks – Sensitive to Outliers

**Natural Language Processing**

**NLP02:**

import nltk

nltk.download()

paragraph contains multiple paragraph

sentences = nltk.sent\_tokenize(paragraph)

words = nltk.word\_tokenize(paragraph)

from nltk.stem import PorterStemmer

from nltk.corpus import stopwords

from nltk.stem import WordNetLematizer

from sklearn.feature\_extraction.text impot CountVectorizer

from sklearn.feature\_extraction.text import TfidfVectorizer

from gensim.models import word2vec

**NLP03:** [**Natural Language Processing| Stemming And Lemmatization Intuition**](https://www.youtube.com/watch?v=JpxCt3kvbLk&list=PLZoTAELRMXVMdJ5sqbCK2LiM0HhQVWNzm&index=3)

Stemming and Lemmatization

* Stemming

Process of reducing Infected words to their word STEM

* Lemmatization

Lemmatization gives meaningful words when compared to Stemming and, Lemmatization take more processing time than Stemming.

**NLP04:** [**Natural Language Processing|Stemming**](https://www.youtube.com/watch?v=1OMmbtVmmbg&list=PLZoTAELRMXVMdJ5sqbCK2LiM0HhQVWNzm&index=4)

from nltk.stem import PorterStemmer

from nltk.corpus import stopwords

sentences = nltk.sent\_tokenize(paragraph)

stemmer = PorterStemmer()

for i in range(len(sentences)):

words = nltk.word\_tokenize(sentence[i])

words = [stemmer.stem(word) for word in words if word not in set(stopwords.words(‘english’))]

sentences[i] = ‘ ‘.join(words)

**NLP05:** [**Natural Language Processing|Lemmatization**](https://www.youtube.com/watch?v=cqcUk6hC5hk&list=PLZoTAELRMXVMdJ5sqbCK2LiM0HhQVWNzm&index=5)

from nltk.stem import WordNetLematizer

**NLP06:** [**Natural Language Processing|Bag Of Words Intuition**](https://www.youtube.com/watch?v=IKgBLTeQQL8&list=PLZoTAELRMXVMdJ5sqbCK2LiM0HhQVWNzm&index=6&pbjreload=101)

Bag of Words – is a process of collecting the frequency of words after ‘stopkey words’ and convert into vectors for to identify independent and dependent feature for further analysis.

|  |  |  |
| --- | --- | --- |
| Sent1 He is a good boy  Sent2 She is a good girl  Sent3 Boy and Girl are good | After implementing Stopkey Words | Sent1 good boy  Sent2 good girl  Sent3 boy girl good |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Feature 1  Good | Feature 2  boy | Feature 3  girl | Dependent feature |
| Sent1 | 1 | 1 | 0 |  |
| Sent2 | 1 | 0 | 1 |  |
| Sent3 | 1 | 1 | 1 |  |

**NLP07:** [**Natural Language Processing|BagofWords**](https://www.youtube.com/watch?v=iu2-G_5YkEo&list=PLZoTAELRMXVMdJ5sqbCK2LiM0HhQVWNzm&index=7)

from sklearn.feature\_extraction.text impot CountVectorizer

**NLP08:** [**Natural Language Processing|BagofWords**](https://www.youtube.com/watch?v=D2V1okCEsiE&list=PLZoTAELRMXVMdJ5sqbCK2LiM0HhQVWNzm&index=8)

Term Frequency (TF) =

Inverse Document Frequency = )

|  |  |
| --- | --- |
| Sent1 good boy  Sent2 good girl  Sent3 boy girl good | Words Frequency  Good. 3  Boy. 2  Girl 2 |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| TF | | | | IDF | | TF \* IDF | F1 | F2 | F3 | o/p |
|  | Sent1 | Sent2 | Sent3 | words | IDF |  | good | boy | girl |  |
| good | 1/2 | 1/2 | 1/3 | good | = 0 | Sent1 | 0 |  | 0 |  |
| boy | 1/2 | 0 | 1/3 | boy |  | Sent2 | 0 | 0 |  |  |
| girl | 0 | 1/2 | 1/3 | girl |  | Sent3 |  |  |  |  |

**NLP09:** [**Natural Language Processing|TF-IDF for Machine Learning| Text Prerocessing**](https://www.youtube.com/watch?v=z9myrLOF_1M&list=PLZoTAELRMXVMdJ5sqbCK2LiM0HhQVWNzm&index=9)

from sklearn.feature\_extraction.text import TfidfVectorizer

**NLP10: Implementing a Spam classifier in python| Natural Language Processing**

import pandas as pd

messages = pd.read\_csv(SMSSpamColletion, sep='\t' names = ['label','message'])

import re

import nltk

nltk.download('stopwords')

from nltk.corpus import stop words

from nltk.stem.porter import PorterStemmer

ps = PorterStemmer()

corpus = []

for I in range (0,len(messages)):

review = re.sub('[^a-zA-Z]',' ',messages['message'][I])

review = review.lower()

review = review.split()

review = [ps.stem(word) for word in review if not word in stopwords.words('engross')]

review = ' '.join(review)

corpus.append(review)

from spleen.feature\_extraction.text import CountVectorizer

cv = CountVectorizer(max\_feature =2500)

X = cv.fit\_transform(corpus).toarray()

y=pd.get\_dummies(messages['label]')

Y = y.iloc[:1].values

from sklearn.model\_selection import train\_test\_split

X\_train,X\_test,Y\_trainm,y\_test = train\_test\_split(X,y,test\_size = 0.20,random\_status =0)

from sklearn.naive\_bayes import MultinomialNB

spam\_detect\_model = MultinomialNB().fit(X\_train,y\_train)

y\_pred=spam\_detect\_model.predict(X\_test)

from sklearn.metrics import confusion\_matrix

confusion\_m = confusion\_matrix(y\_test,y\_pred)

from sklean.metrics import accuracy\_score

accuracy = accuracy\_score(y\_test,y\_pred)

**NLP11: Word2Vec Easily Explained- Data Science**

from gensim.models import word2vec

**NLP12: Kaggle Competition- Predict Stock Price Movement Based On News Headline using NLP**

**NLP13: FAKE NEWS CLASSIFIER WITH MACHINE LEARNING ALGORITHMS USING Natural Language Processing- PART1**

**NLP14: FAKE NEWS CLASSIFIER WITH MACHINE LEARNING ALGORITHMS USING Natural Language Processing- PART2**

**NLP15:** [**Tutorial 29- Why Use Recurrent Neural Network and Its Application**](https://www.youtube.com/watch?v=CPl9XdIFbYA&list=PLZoTAELRMXVMdJ5sqbCK2LiM0HhQVWNzm&index=15)

**NLP16:** [**Tutorial 30- Recurrent Neural Network Forward Propogation With Time**](https://www.youtube.com/watch?v=u8utlK_c5C8&list=PLZoTAELRMXVMdJ5sqbCK2LiM0HhQVWNzm&index=16)

>

loss = ( y^ - y)

**NLP17:** [**Tutorial 31- Back Propagation In Recurrent Neural Network**](https://www.youtube.com/watch?v=6EXP2-d_xQA&list=PLZoTAELRMXVMdJ5sqbCK2LiM0HhQVWNzm&index=17)

**NLP18:** [**Tutorial 32- Problems In Simple Recurrent Neural Network**](https://www.youtube.com/watch?v=mDaEfPgwtgo&list=PLZoTAELRMXVMdJ5sqbCK2LiM0HhQVWNzm&index=18)

**NLP19:** [**Tutorial 32- Problems In Simple Recurrent Neural Network**](https://www.youtube.com/watch?v=mDaEfPgwtgo&list=PLZoTAELRMXVMdJ5sqbCK2LiM0HhQVWNzm&index=18)

**NLP20:** [**Word Embedding - Natural Language Processing| Deep Learning**](https://www.youtube.com/watch?v=pO_6Jk0QtKw&list=PLZoTAELRMXVMdJ5sqbCK2LiM0HhQVWNzm&index=20)

cosine similarity helps to find relationship between two vectors

What is low dimension and dense vector?

**NLP21:** [**Implementing Word Embedding Using Keras- NLP | Deep Learning**](https://www.youtube.com/watch?v=TsXR7_vtusQ&list=PLZoTAELRMXVMdJ5sqbCK2LiM0HhQVWNzm&index=21)

**NLP22:** [**Kaggle Faker News Classifier Using LSTM- Deep LEarning| Natural Language Processing**](https://www.youtube.com/watch?v=MXPh_lMRwAI&list=PLZoTAELRMXVMdJ5sqbCK2LiM0HhQVWNzm&index=22)

**NLP23:** [**Stock Price Prediction And Forecasting Using Stacked LSTM- Deep Learning**](https://www.youtube.com/watch?v=H6du_pfuznE&list=PLZoTAELRMXVMdJ5sqbCK2LiM0HhQVWNzm&index=23)

**NLP24:** [**Bidirectional RNN Indepth Intuition- Deep Learning Tutorial**](https://www.youtube.com/watch?v=D-a6dwXzJ6s&list=PLZoTAELRMXVMdJ5sqbCK2LiM0HhQVWNzm&index=24)

What is the difference between Bidirectional RNN and RNN Backward Propagation

**NLP25:** [**Sequence To Sequence Learning With Neural Networks| Encoder And Decoder In-depth Intuition**](https://www.youtube.com/watch?v=jCrgzJlxTKg&list=PLZoTAELRMXVMdJ5sqbCK2LiM0HhQVWNzm&index=25)

**NLP26:** [**Encoder And Decoder- Neural Machine Learning Language Translation Tutorial With Keras- Deep Learning**](https://www.youtube.com/watch?v=f-JCCOHwx1c&list=PLZoTAELRMXVMdJ5sqbCK2LiM0HhQVWNzm&index=26)

**NLP27:** [**Problems With Encoders And Decoders- Indepth Intuition**](https://www.youtube.com/watch?v=tHf4CmTH1QE&list=PLZoTAELRMXVMdJ5sqbCK2LiM0HhQVWNzm&index=27)

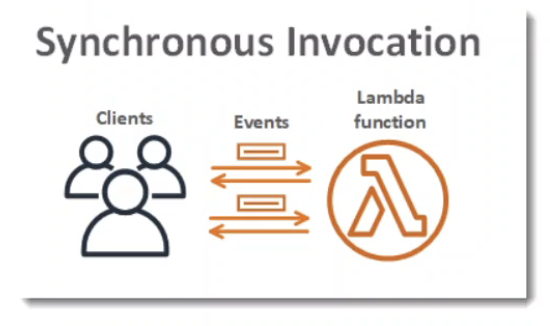
**NLP28:** [**Live Session- Understanding Attention Models Architecture And Maths Intuition- Deep Learning**](https://www.youtube.com/watch?v=fdhojC37_Co&list=PLZoTAELRMXVMdJ5sqbCK2LiM0HhQVWNzm&index=28)

**Gen05: Difference Between fit(), transform(), fit\_transform() and predict() methods in Scikit-Learn**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| EDA | Feature Engineering | Feature Selection | Model Creation | Model Deployment |
| **Data Pre-Processing**  Transformers  \* Standard Scalar – Standard Normal Distirubtion  \* MinMax Scalar  \* PCA – dimensionality reduction  \* One Hot Encoding  \* Remove Nulls - Imputer  In this section the each data set are fit to the transformer before it get into Transformers  In Sciket fit\_transformers -> transformers | | | **Model Training**  Model Selection  Linear Regression  Decision Tree  Logistic Regression  Random Forest  AdaBoost  Gradient Decent  XgBoost  In Model we do 1. **FIT and TRAIN in case of Training,**  2.Prediction with Test / New data  For Training use fit\_transform(x\_train\_data)  For Test use transform(x\_test\_data) |  |

Types of sampling techniques in statistics?

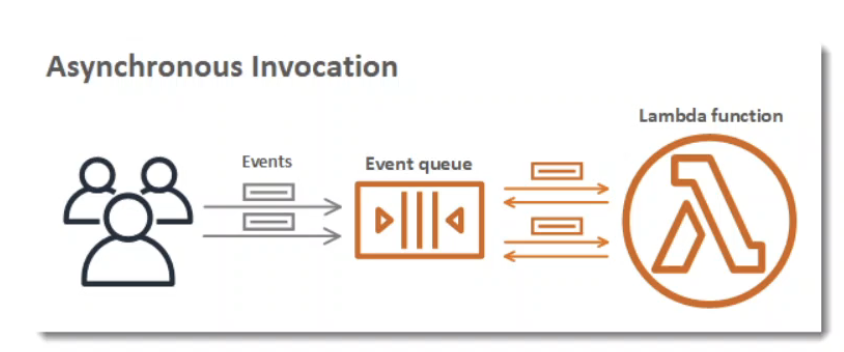
Random, Systematic, Clustered, Stratified, Convenient



What is Synchronous Invocation

- with synchronous invocation you wait for the function to process the event and return a response

- Synchronous invocations are best suited for Machine Learning workflow



What is Asynchronous Invocation

- with asynchronous invocation, Lambda queues the event for processing, so you don't have to wait for a response from Lambda

- For asynchronous invocation, Lambda handles retries and can send invocation records to a destination

Feature Engineering

1. EDA - Exploratory Data Analysis

Raw Data:

1. How many Numerical Feature (Histogram, PDF), Continues, Discreate

2. How many Categorical Feature. with cardinality (number of unique category and how many of them, Rare categorical feature (category < 1% of total category of the feature)

3. Missing Values

4. Outliers

5. Cleaning

6. Relationship between dependent and independent variable

2. Handling Missing Values

3. Handling Imbalanced Data set

4. Treating outlier

5. Scaling the data

6. Converting Categorical feature into numerical features

Feature Selection

1. Correlation

2. Chi squre

3. Feature Importance

4. K Neihbour